Implementation of DFS

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51 | **import** **java.util.\***;  **class** **MyGraph** {  Map<Integer,HashSet<Integer>> mp;  **public** **MyGraph**(){  mp = **new** HashMap<>();  }  **public** **void** **addEdge**(**int** v1,**int** v2,**boolean** isBiDir){  HashSet<Integer> v1Neighbor = mp.getOrDefault(v1,**new** HashSet<>());  v1Neighbor.add(v2);  mp.put(v1,v1Neighbor);  **if**(isBiDir) addEdge(v2,v1,**false**);  }  **public** **void** **display**(){  **for**(Map.Entry<Integer,HashSet<Integer>> res : mp.entrySet()){  System.out.println(res.getKey() + " -> "+ res.getValue());  }  }  **public** **void** **dfs**(**int** src){  Stack<Integer> DFS = **new** Stack<>();  DFS.push(src);  HashSet<Integer> vis = **new** HashSet<>();  vis.add(src);  **while** (!DFS.isEmpty()){  **int** temp = DFS.pop();  System.out.print(temp+" ");  HashSet<Integer> child = mp.get(temp);  **for**(**int** tem : child){  **if**(!vis.contains(tem)){  vis.add(tem);  DFS.push(tem);  }  }  }  System.out.println();  }  }  **public** **class** **Graph\_with\_Map** {  **public** **static** **void** **main**(String[] args) {  MyGraph obj = **new** MyGraph();  obj.addEdge(**1**,**2**,**true**);  obj.addEdge(**1**,**3**,**true**);  obj.addEdge(**3**,**4**,**false**);  obj.addEdge(**3**,**5**,**true**);  obj.addEdge(**5**,**6**,**true**);  obj.addEdge(**2**,**4**,**true**);  obj.display();  obj.dfs(**1**);  }  } |